RL for Robot Foundation Models

CS 224R

Course reminders

- Poster session next Wednesday
- Final report due following Monday

^ no extensions or late days

Plan for today

Last lecture: Can we do RL *on simulated robots* and transfer behaviors to the real world?

Today: How to do RL *on real robots* with pretrained foundation models?

Plan for today

Today: How to do RL *on real robots* with pretrained foundation models?

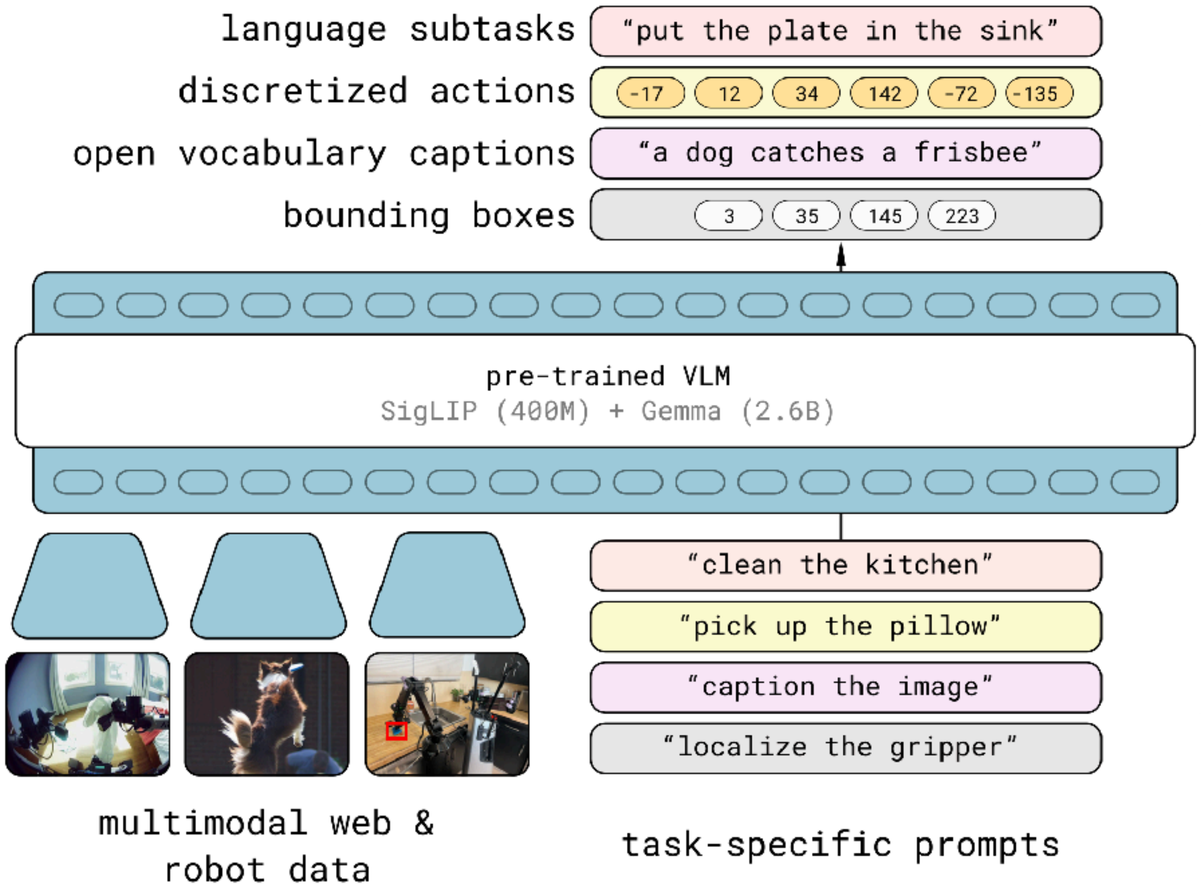
1. What's the problem
    a. Promise & opportunities from robot foundation models
    b. Why these models are hard to train with RL
2. Key design tools
    a. Can we just use PPO?
    b. Offline RL: Can we just use supervised learning?
    c. Online RL:
        a. Reducing dimensionality
        b. Learning residuals or edits

Caveat: This is an open, active research problem!

Today: cover some recent themes and my opinion on the area.

Robot foundation models

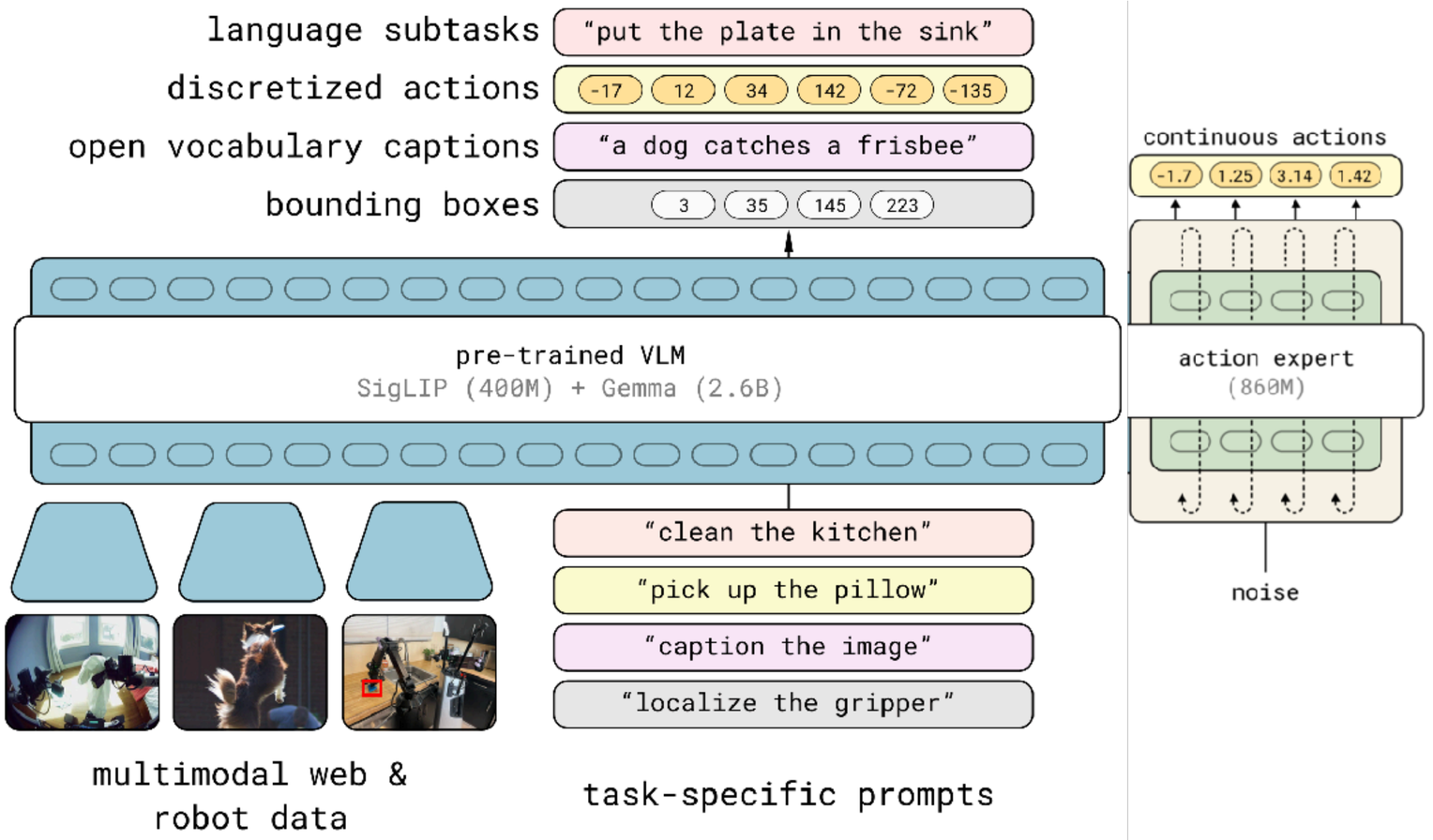
Most commonly: a vision-language-action (VLA) model



- start with pretrained vision-language model (VLM)
- alternative design: start with generative video model
- training data mixture often includes:
  - robot demonstration data (imitation learning)
  - VLM tasks (question answering, captioning, detection)
  - human video data (motion prediction)
- often includes an diffusion-based "action expert"

Robot foundation models

Most commonly: a vision-language-action (VLA) model



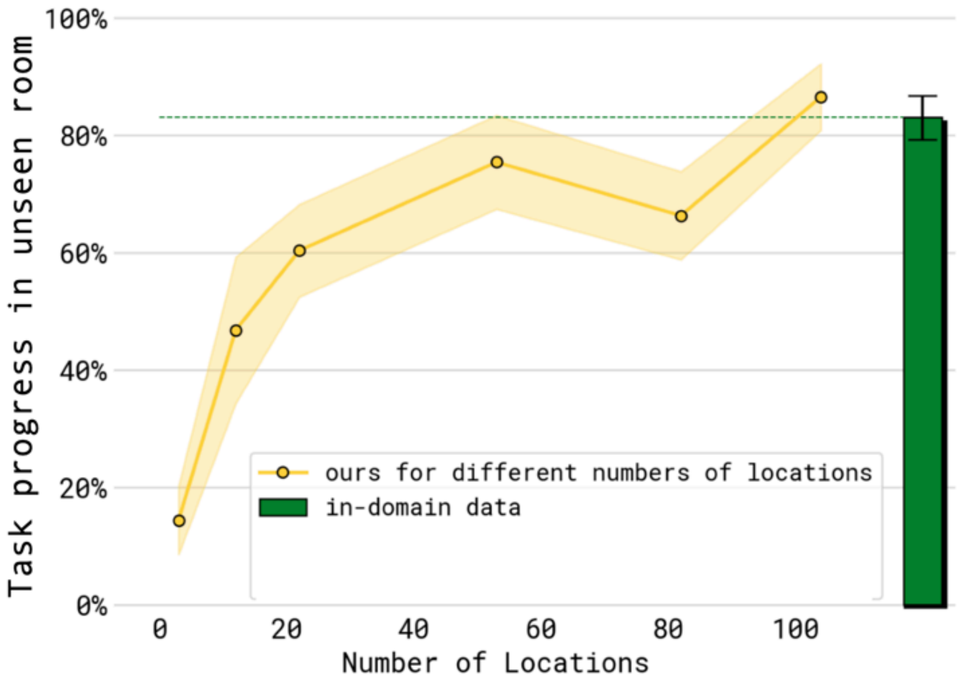
Action expert:

- uses diffusion or flow matching to predict continuous actions
- attends to all activations of LLM backbone
- designed to avoid multiple forward passes through entire backbone
- gradient is often not passed to backbone

What is the problem with VLAs?

Trained with *imitation learning* -> Performance often plateaus at around 80%

$\pi_{0.5}$ performance in unseen rooms



- similar to how LLMs become more performant with RL after SFT
- for robots to act *autonomously*, often need 99%+ reliability
  - -> natural use-case for RL-fine-tuning!
- pretrained VLA can serve as effective initialization for RL

Note: DAgger can also be helpful, often used in conjunction with RL

Why RL for VLAs is hard

1. VLAs are **large**: gradient updates are **computationally expensive**

- often want to do training in the cloud, with robot roll-outs done on local computer
- longer iteration time for any single experiment
- extensive hyperparameter tuning is expensive, time-consuming

2. VLA is **pre-trained with imitation learning**

- no pre-trained value function or critic
- often trained with diffusion / flow matching -> harder to apply RL algs out of the box
- often trained with action chunking -> also an uncommon choice for RL

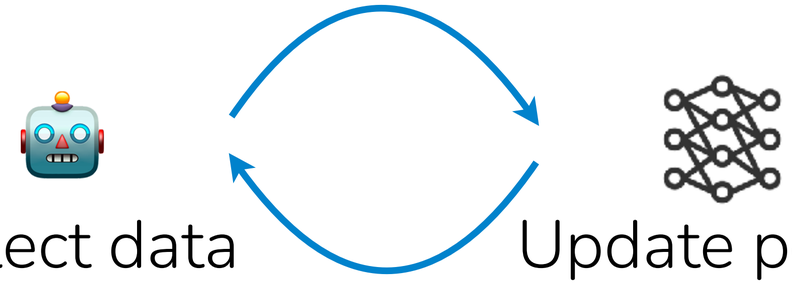
Online vs. Offline RL for VLAs

online RL
(e.g. SAC)

Collect data
1 timestep of data

Update policy
1 gradient step

(iterated) offline RL



Collect data
1k episodes

Update policy
10k gradient steps

If you have a bug or messed up a hyperparameter (e.g. learning rate, # epochs, grad clipping, etc...):

- Online RL: need to rerun experiment, *recollect data*
- Offline RL: rerun training on existing dataset

Much simpler for large models!

Plan for today

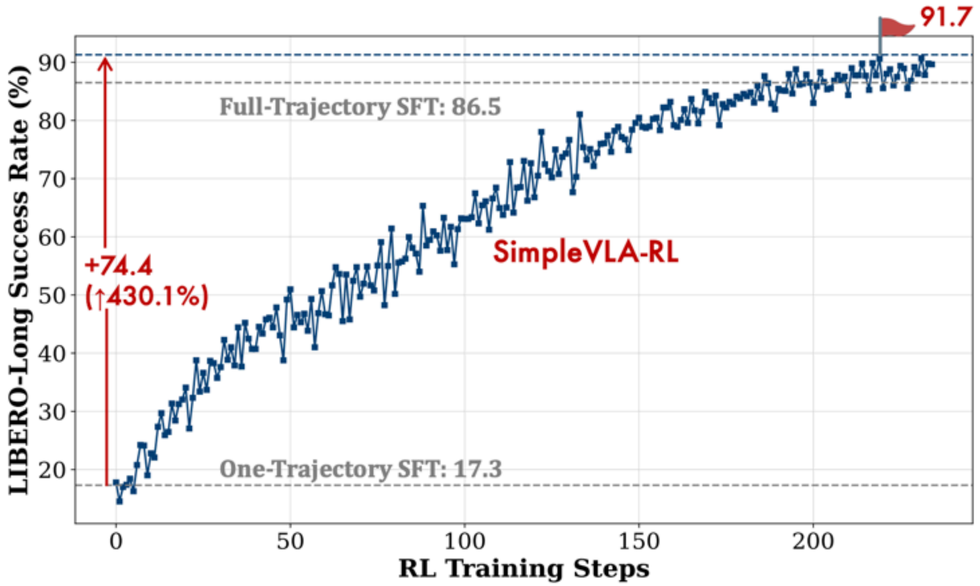
Today: How to do RL *on real robots* with pretrained foundation models?

1. What's the problem
    a. Promise & opportunities from robot foundation models
    b. Why these models are hard to train with RL
2. **Key design tools**
    a. Can we just use PPO?
    b. Offline RL: Can we just use supervised learning?
    c. Online RL:
        a. Reducing dimensionality
        b. Learning residuals or edits

Caveat: This is an open, active research problem!

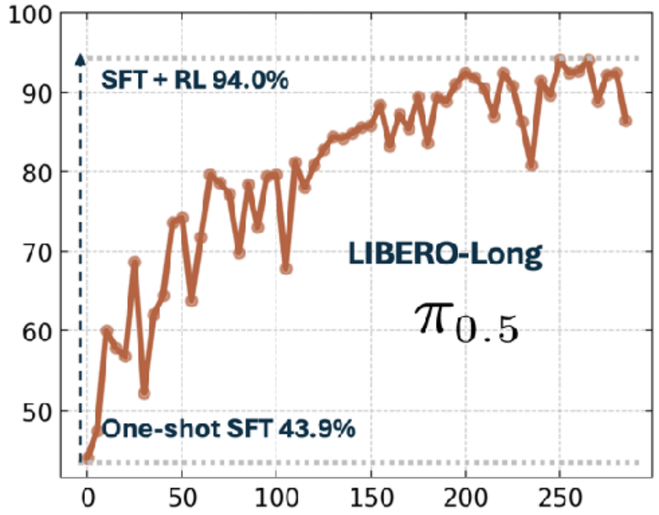
Can we just do PPO?

Finetuning OpenVLA with RL



Li et al. SimpleVLA-RL. 2025

Finetuning $\pi_{0.5}$ with RL



Chen et al. π_{RL}. 2026

Yes, but: requires massive amount of online policy roll-outs

(many papers don't even report # of samples!)

results are limited to simulation-based training

Let's start with offline RL

^
iterated

Key theme #1: Can we formulate a method based on supervised learning?

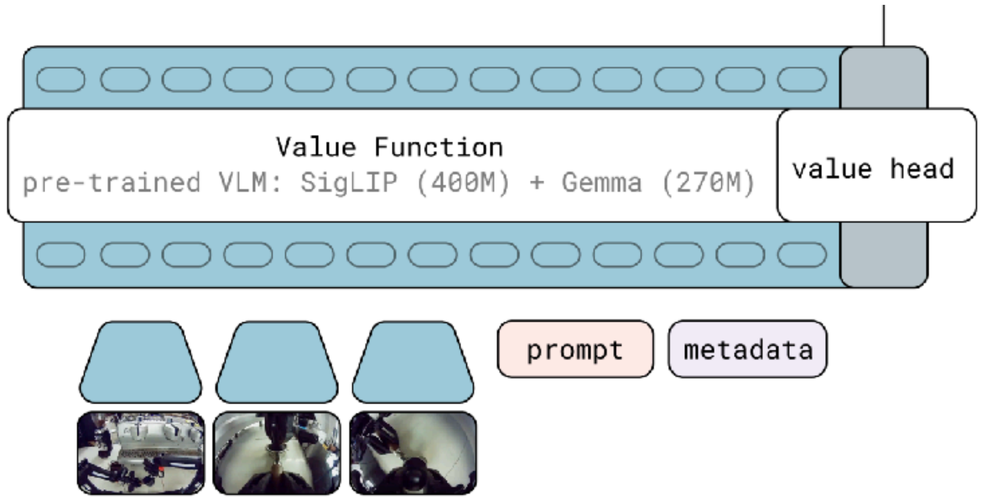
-> If so, this may be easier to scale to large models and datasets!

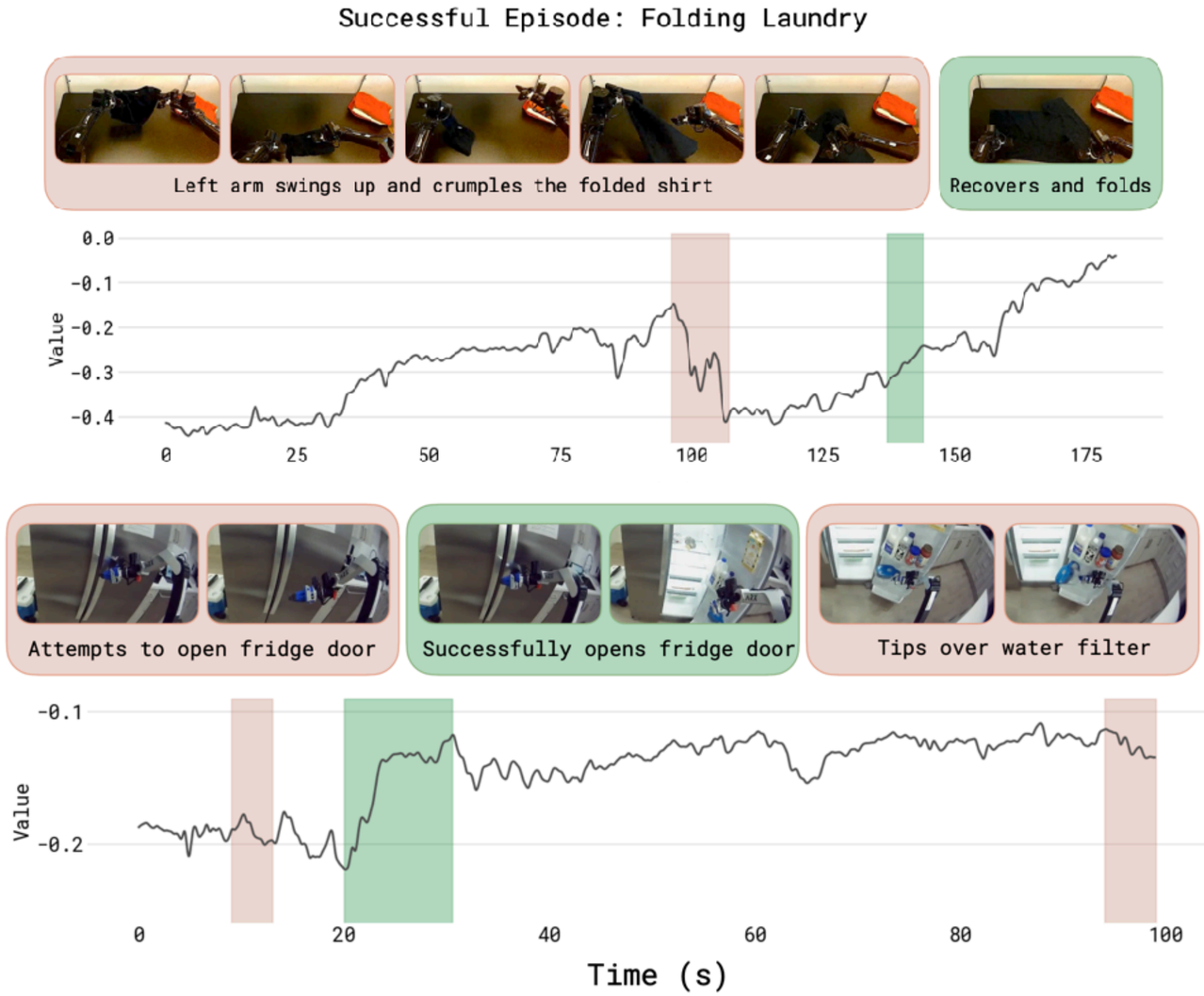
Two parts:

1. Learn a value function <- can fit V^π using Monte Carlo
2. Use that value function to get a better policy <- supervise policy to take actions that V^π thinks is better

Supervised Value Function Training (i.e. Monte Carlo)

Fitting a multi-task, language-conditioned V^{π} on a large-scale demo dataset



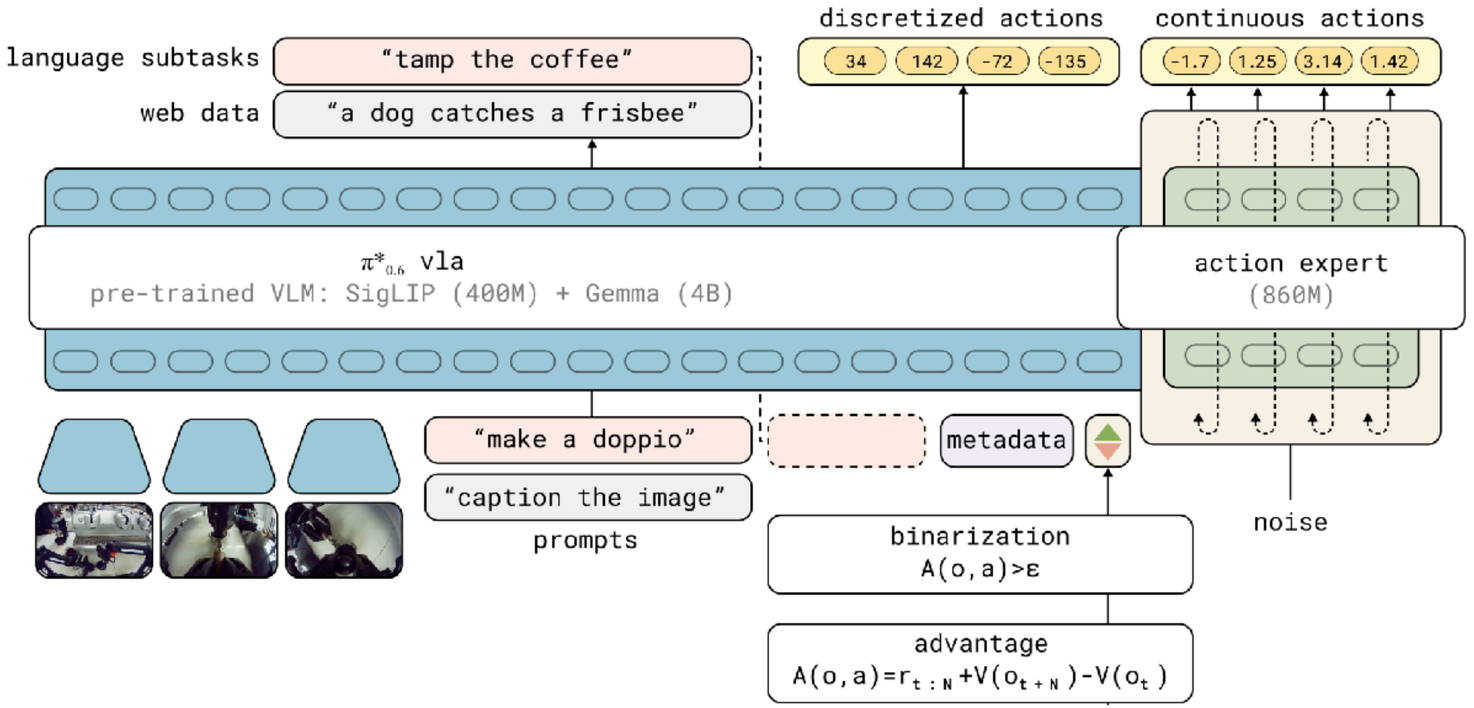


- Use pre-trained VLM
- Conditioned on current images, language prompt, episode metadata
- Predict time to go

Physical Intelligence. *π*0.6 : a VLA That Learns From Experience*. 2025.

Supervised Policy Training

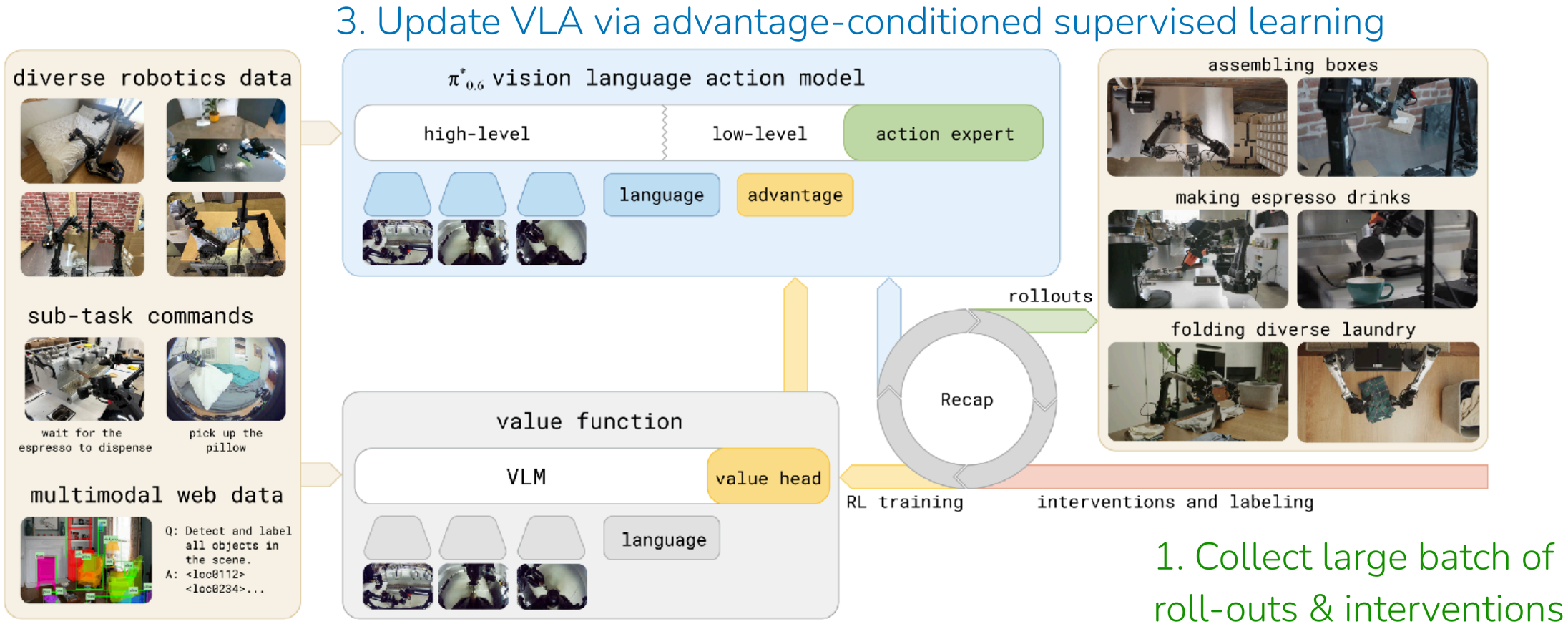
Train policy with advantage conditioned supervised learning



- Estimate *advantage A(s,a)* using predicted values
- Binarize the advantage to tell the policy if the action is good or bad
- Fine-tune the policy with supervised learning, conditioned on binarized advantage

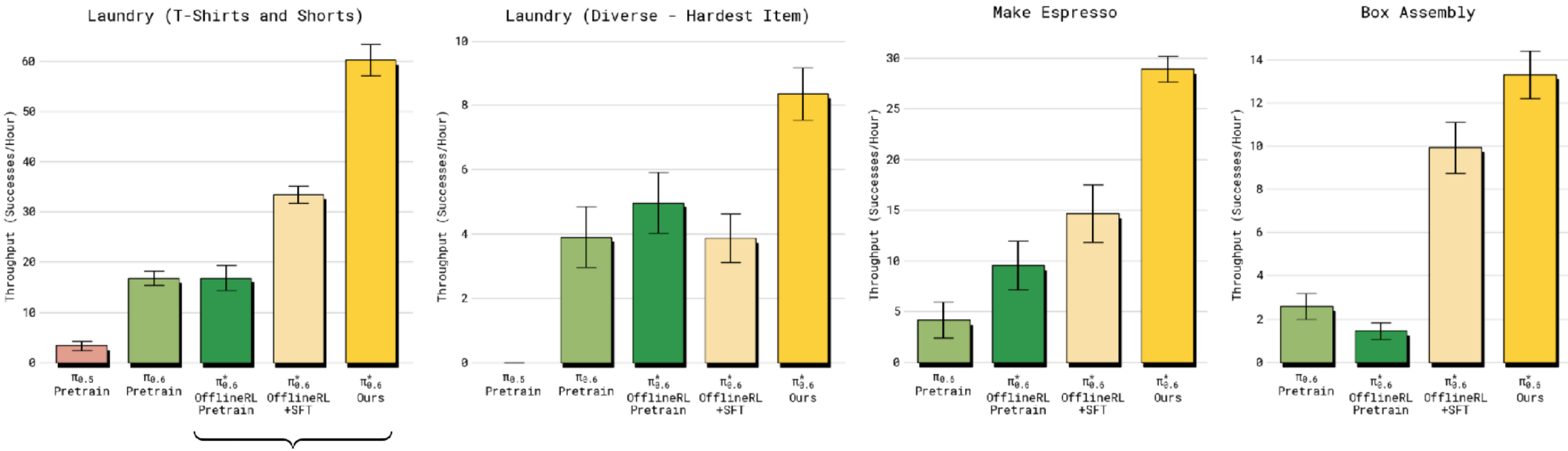
Physical Intelligence. *π*0.6 : a VLA That Learns From Experience*. 2025.

Full algorithm



Physical Intelligence. *π*0.6 : a VLA That Learns From Experience*. 2025.

Does RL improve over the base VLA?



Phases of training

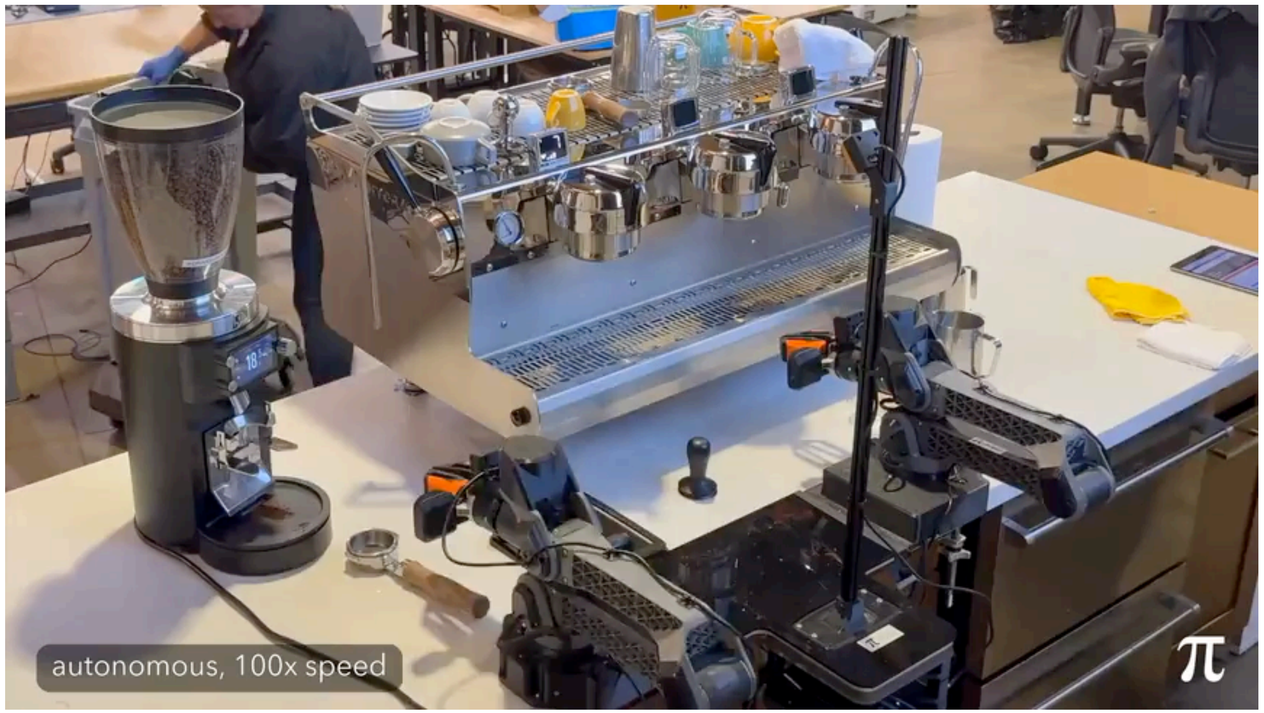
2x improvement in throughput from RL post-training compared to IL post-training

Physical Intelligence. *π*0.6 : a VLA That Learns From Experience*. 2025.[16]

Robots doing cool stuff

Making a latte in collaboration with a person

Making lattes reliably



13 hours of operation

Is this the best recipe for VLA RL?

1. TD updates should be able to improve over MC value learning, even in large-scale setting.
2. Should also benefit from more powerful policy improvement methods
3. Online RL should be more data efficient, reach higher performance by more quickly seeking out failure modes, ruling out new strategies
   (but with more infrastructural complexity)

Plan for today

Today: How to do RL *on real robots* with pretrained foundation models?

1. What's the problem
    a. Promise & opportunities from robot foundation models
    b. Why these models are hard to train with RL
2. Key design tools
    a. Can we just use PPO?
    b. Offline RL: Can we just use supervised learning?
    c. **Online RL**:
        a. Reducing dimensionality
        b. Learning residuals or edits

Caveat: This is an open, active research problem!

Can we do online RL for VLAs?

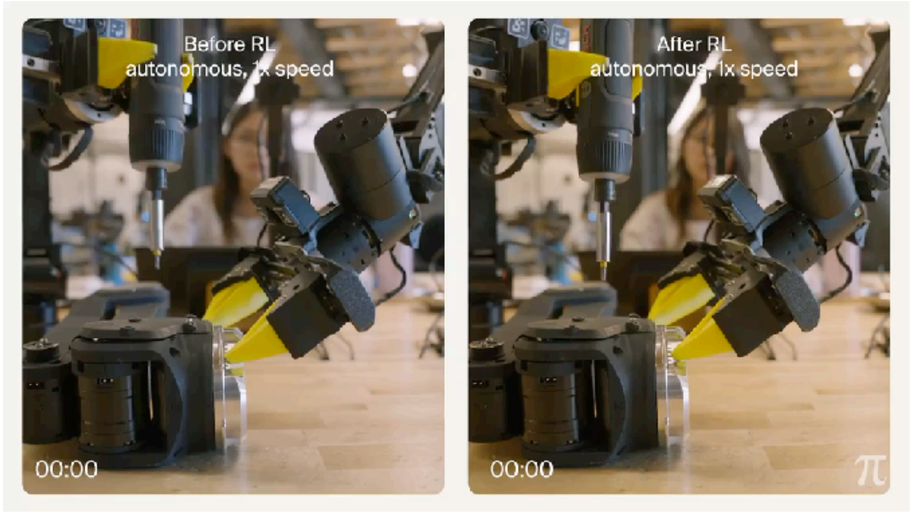
Can we improve upon the VLA without fine-tuning it end-to-end?

Key theme #2: Learn a separate Gaussian policy using the VLA's representation.

Version A: Treat the noise of VLA diffusion as an action space, and train RL policy to control noise.

Wagenmaker et al. DSRL. 2025

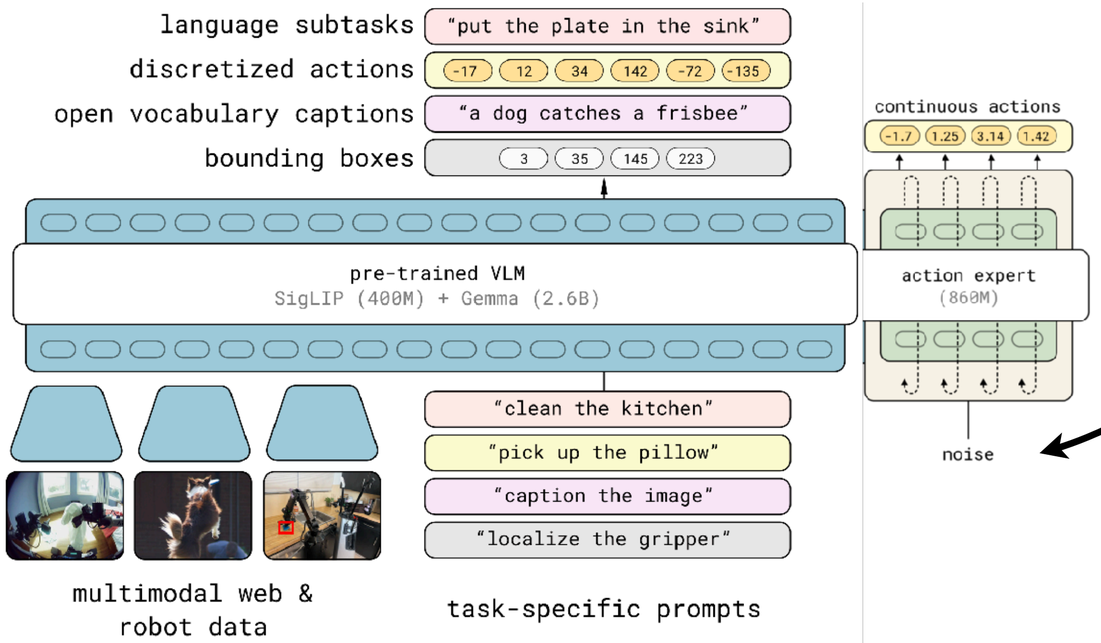
Version B: Compress visual representation of VLA and do RL on top of this representation



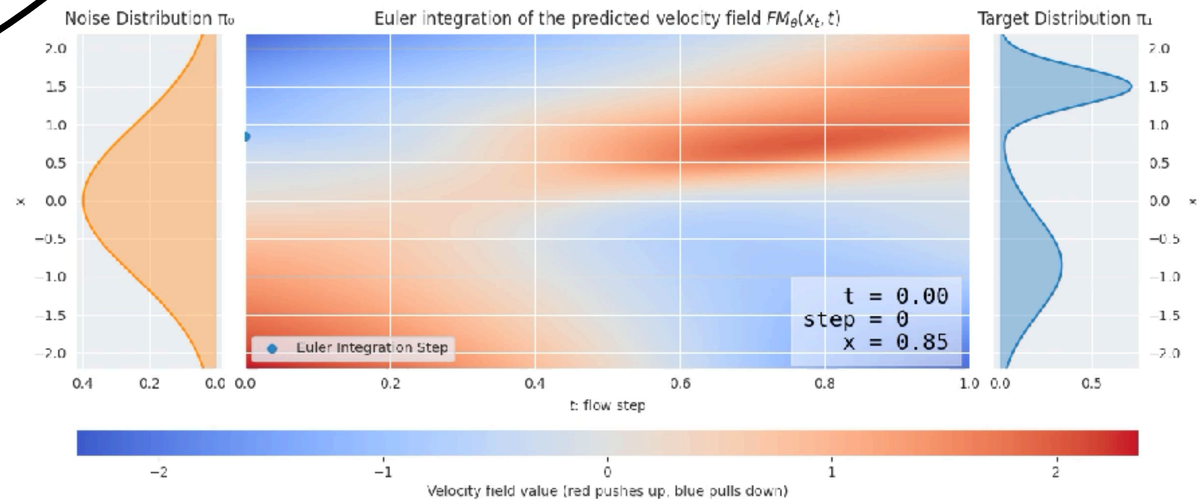
Xu et al. RLT. 2026

Aside: After RL, you can distill policy data back into the VLA!

Can we do online RL for VLAs?



Different noise vectors lead to different actions

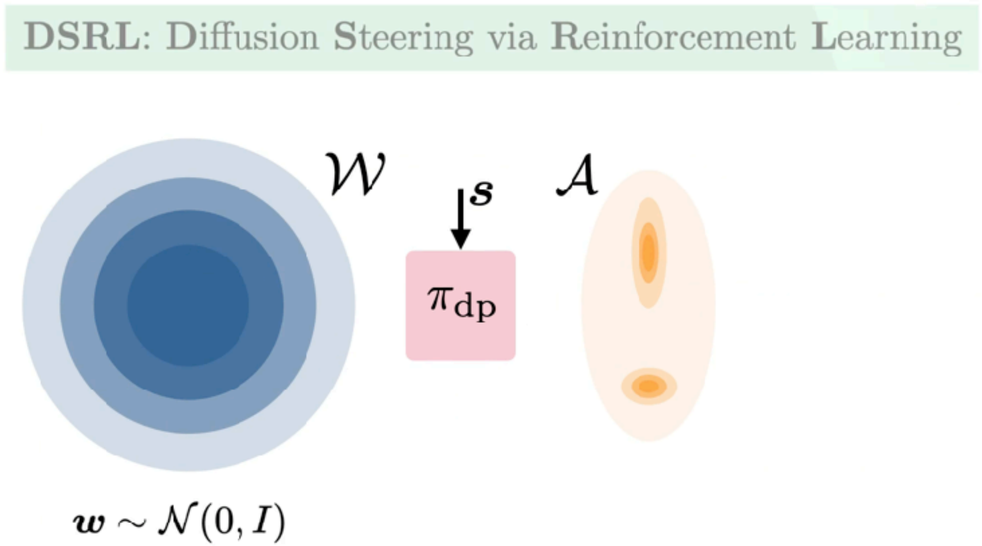


Diffusion steering: train policy to output noise that leads to good actions

Wagenmaker*, Nakamoto*, Zhang* et al. Steering Your Diffusion Policy with Latent Space Reinforcement Learning. CoRL 2025.

Can we do online RL for VLAs?

Diffusion steering: train policy to output noise that leads to good actions



Policy sampling:

1. Sample $\mathbf{w}_t \sim \pi_{steer}(\cdot \,|\, \mathbf{s}_t; \theta)$
2. Denoise $\mathbf{a}_{t:t+h} = \pi_{VLA}(\mathbf{s}_t, \mathbf{w}_t)$
3. Run $\mathbf{a}_{t:t+h}$ in the env, observe $\mathbf{s}_{t+h}$

Training:

1. Sample roll-out $\mathbf{s}_1, \mathbf{w}_1, r_1, \ldots \mathbf{s}_T$, add to buffer
2. Sample minibatch of transitions from buffer
3. Update $Q_\phi(\mathbf{s}_t, \mathbf{w}_t)$ and $\pi_{steer}(\mathbf{w}_t \,|\, \mathbf{s}_t; \theta)$ using SAC

Wagenmaker*, Nakamoto*, Zhang* et al. Steering Your Diffusion Policy with Latent Space Reinforcement Learning. CoRL 2025.

Can we do online RL for VLAs?

Diffusion steering: train policy to output noise that leads to good actions

Task	π_0	DSRL
Turn on toaster	5/20	**18/20**
Put spoon on plate	15/20	**19/20**

Training data: 65 online episodes, ~10k steps

Roughly O(100x) more efficient than PPO

Wagenmaker*, Nakamoto*, Zhang* et al. Steering Your Diffusion Policy with Latent Space Reinforcement Learning. CoRL 2025.

Plan for today

Today: How to do RL *on real robots* with pretrained foundation models?

1. What's the problem
   a. Promise & opportunities from robot foundation models
   b. Why these models are hard to train with RL
2. Key design tools
   a. Can we just use PPO?
   b. Offline RL: Can we just use supervised learning?
   c. Online RL:
      a. Reducing dimensionality
      b. Learning residuals or edits

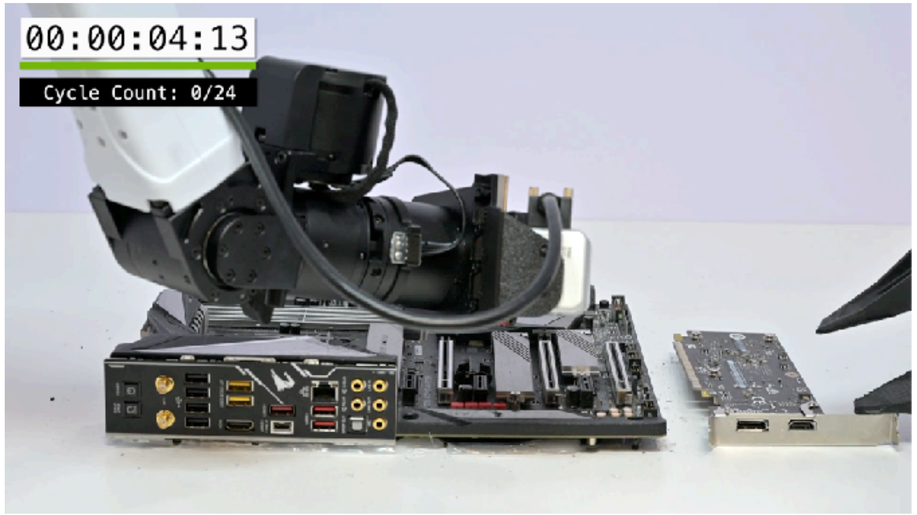
Caveat: This is an open, active research problem!

Can we do online RL for VLAs?

Can we improve upon the VLA without fine-tuning it end-to-end?

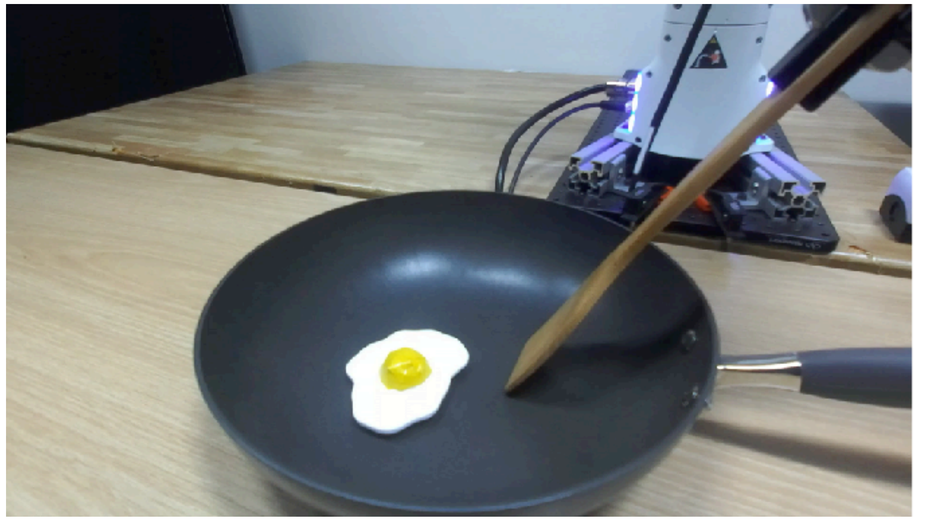
Key theme #3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.

Version A: Actor-critic algorithm + distill back into VLA



Xiao et al. *Probe-Learn-Distill*. 2025

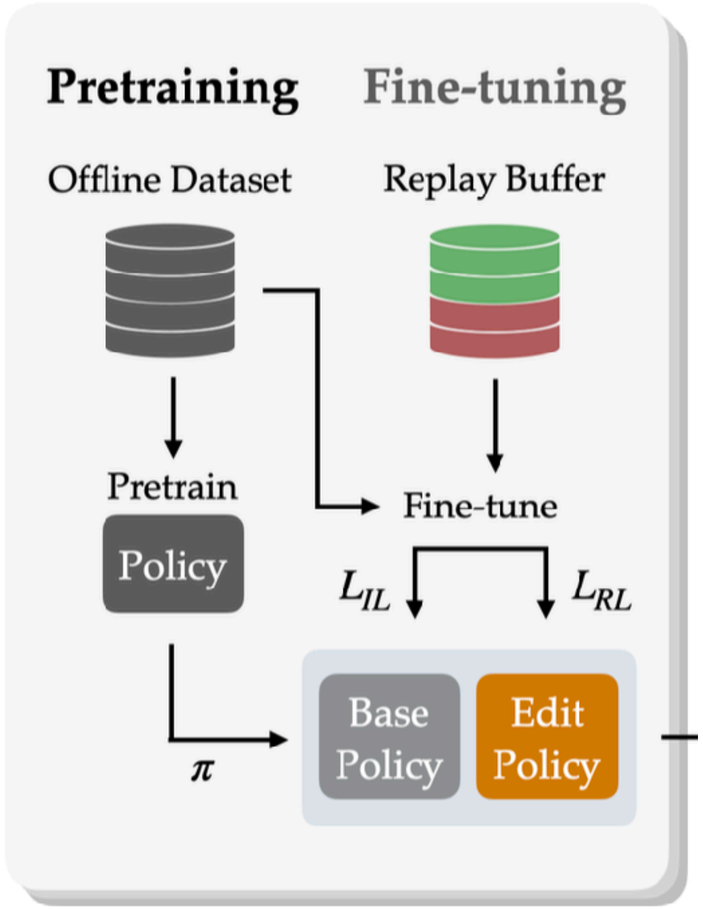
Version B: Actor-critic algorithm + best-of-N sampling at test time



Dong et al. EXPO-FT. 2026

Can we do online RL for VLAs?

Key theme #3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.



1. Optimize smaller Gaussian edit policy to maximize Q-values

2. Base policy trained with imitation on all successes.

Note: On it's own, this may not be stable.

- Edit policy will naturally lag behind Q-function.
- Edit policy could collapse.

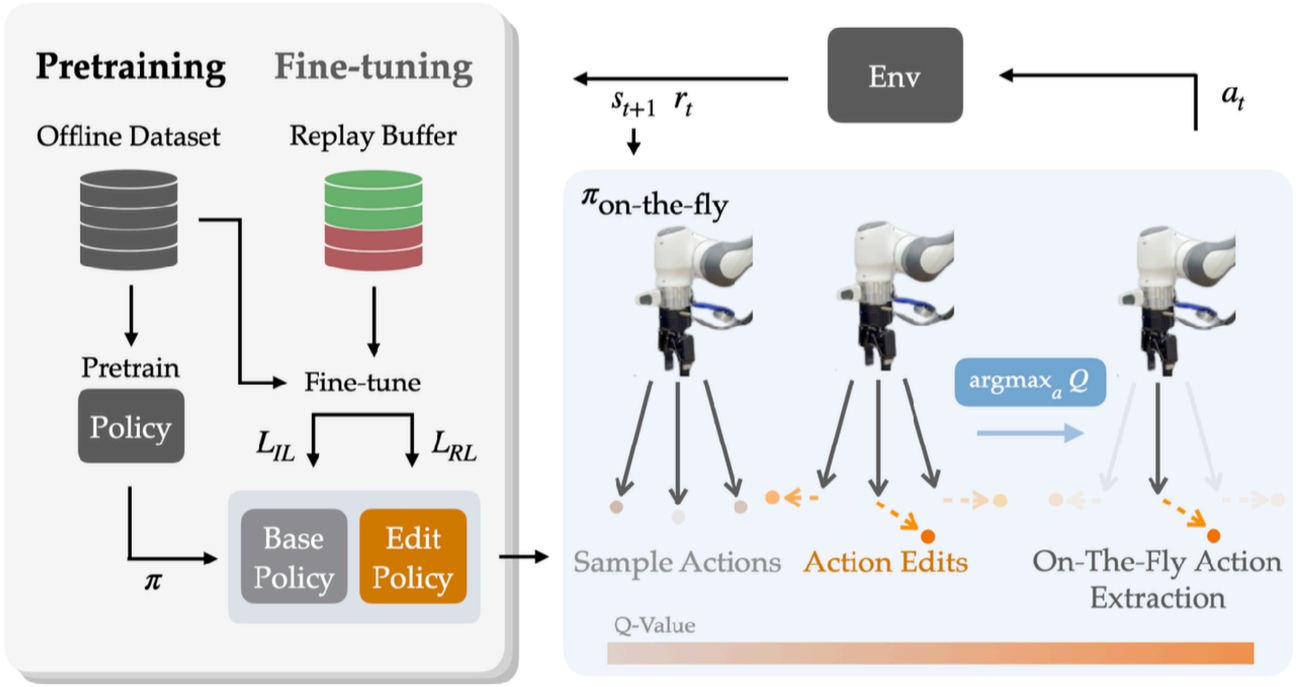
(like typical RL)

Dong, Li, Sadigh, Finn. EXPO: *Stable Reinforcement Learning with Expressive Policies*. ICLR 2026

Can we do online RL for VLAs?

Key theme #3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.

For stability: Maximize latest Q-function on the fly via sampling.



1. Sample multiple a from π_{base}
2. Sample multiple $\tilde{a}$ from $\pi_{\text{edit}}(\tilde{a} \mid s, a)$
3. Pick $\{a_1, \ldots, a_n, \tilde{a}_1, \ldots, \tilde{a}_n\}$ with highest Q

✓ reduces lag with Q-function

✓ resilient to possible edit policy collapse

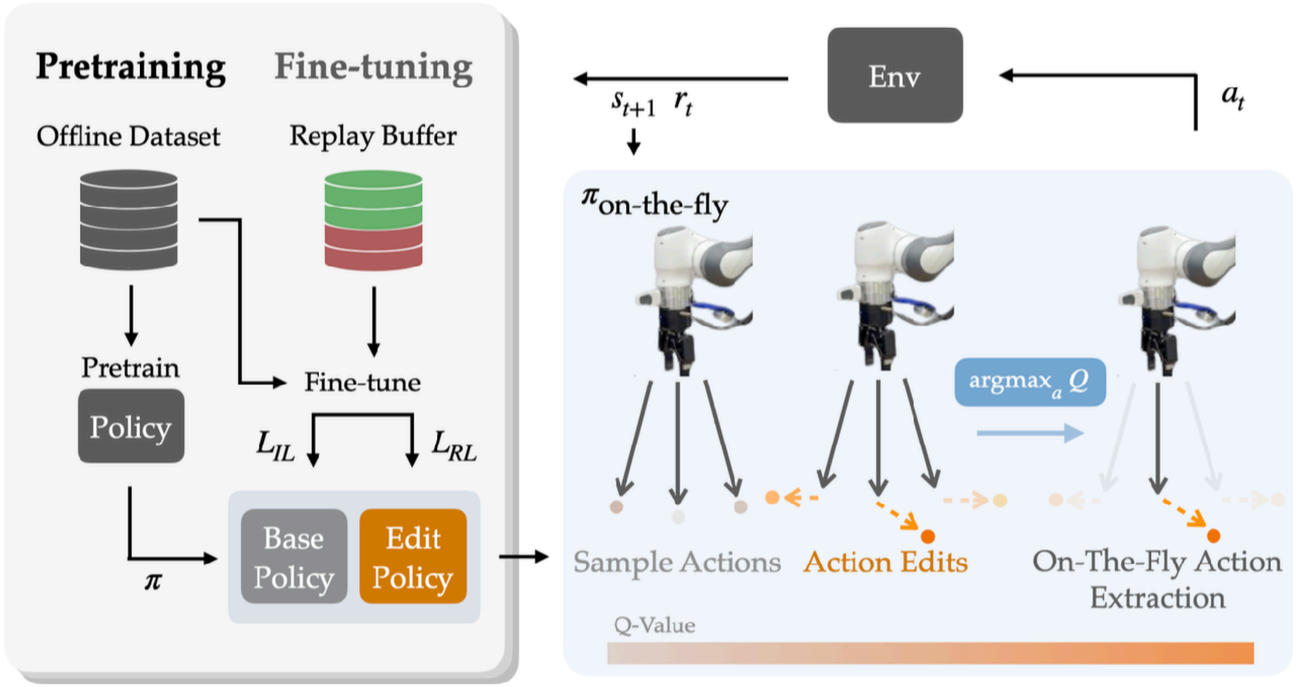
Perhaps viewed as a form of test-time scaling?

Dong, Li, Sadigh, Finn. EXPO: *Stable Reinforcement Learning with Expressive Policies*. ICLR 2026

Can we do online RL for VLAs?

Key theme #3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.

For stability: Maximize latest Q-function on the fly via sampling.



1. Sample multiple a from π_{base}
2. Sample multiple $\tilde{a}$ from $\pi_{\text{edit}}(\tilde{a} \mid s, a)$
3. Pick $\{a_1, \ldots, a_n, \tilde{a}_1, \ldots, \tilde{a}_n\}$ with highest Q

✓ reduces lag with Q-function

✓ resilient to possible edit policy collapse

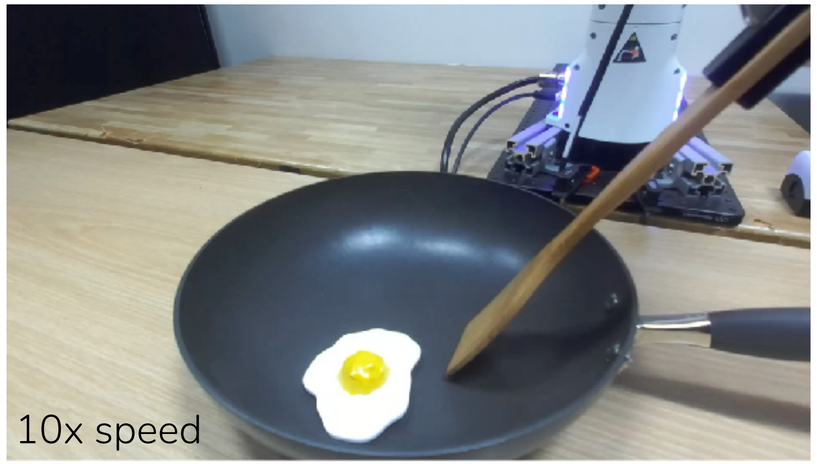
‼ Important note: When fitting Q^π, how to pick actions a' in Bellman backup?

Use on-the-fly policy!

Dong, Li, Sadigh, Finn. EXPO: *Stable Reinforcement Learning with Expressive Policies*. ICLR 2026

Can we do online RL for VLAs?

Key theme #3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.



Task	Success Rate (x/30)					
	SFT	HG-DAgger	DSRL	HIL-SERL	HIL-SERL (M)	EXPO-FT
Egg Flip	16/30	18/30	15/30	13/30	-	**30/30**
Cube Pick	22/30	26/30	24/30	0/30	27/30	**30/30**
Pool Shot	23/30	14/30	25/30	1/30	13/30	**30/30**
Flower Insertion	14/30	24/30	12/30	8/30	-	**30/30**
Average	18.8/30	20.5/30	19/30	5.5/30	20/30	**30/30**

Training data: 19 min of experience on average, ~11k steps

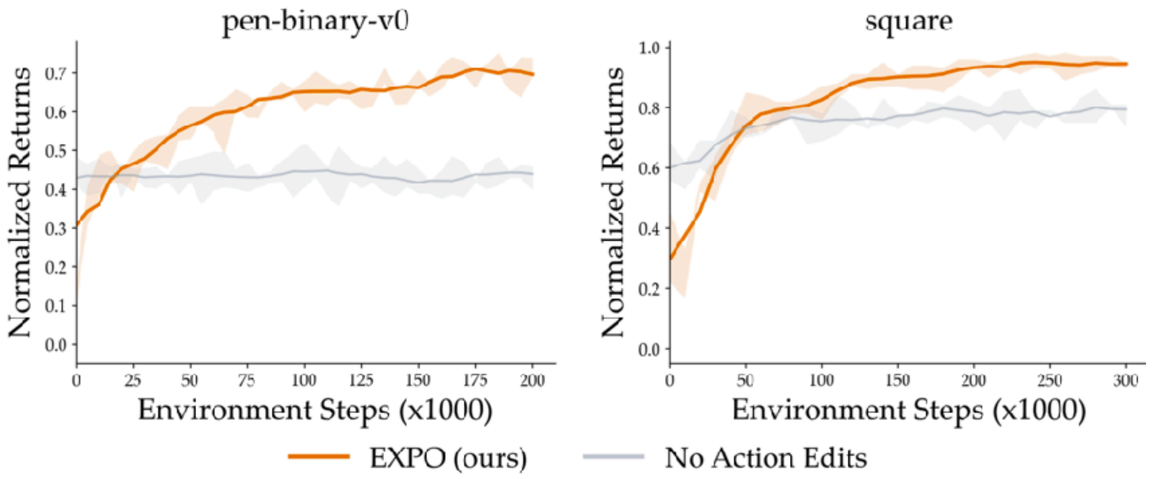
- Higher reliability than SFT, DAgger
- Learns more efficiently & effectively than DSRL, HIL-SERL

Dong*, Hung*, Gao, Sadigh, Finn. EXPO-FT: Sample-Efficient RL Fine-Tuning for VLAs. 2026

Can we do online RL for VLAs?

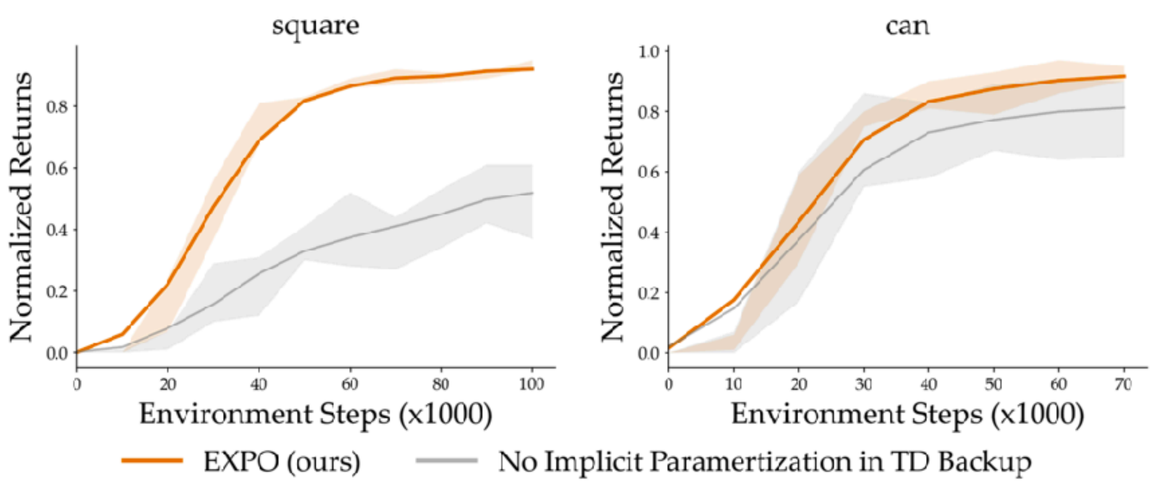
Key theme #3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.

No edit policy?



Value maximization hindered significantly

Don't use on-the-fly policy in Bellman backup?



Poor performance in some environments

Dong, Li, Sadigh, Finn. EXPO: *Stable Reinforcement Learning with Expressive Policies*. ICLR 2026

Plan for today

Today: How to do RL *on real robots* with pretrained foundation models?

1. What's the problem
    a. Promise & opportunities from robot foundation models
    b. Why these models are hard to train with RL
2. Key design tools
    a. Can we just use PPO?
    b. Offline RL: Can we just use supervised learning?
    c. Online RL:
        a. Reducing dimensionality
        b. Learning residuals or edits

Caveat: This is an open, active research problem!

Summary

Challenges: VLAs are **large**: gradient updates are **computationally expensive**

VLAs are **pre-trained with imitation learning**

Key themes:

#1: Formulate a method based on supervised learning for scalability *(Offline RL)*

#2: Learn a separate Gaussian policy using the VLA's representation. *(Online RL)*

#3: Learn a small Gaussian policy that edits the (diffusion) VLA's actions.

Outlook on RL for VLAs

1. Exciting progress!

- evidence that you can substantially improve performance, speed of state-of-the-art VLAs using RL
- evidence of getting to performance needed for real-world deployment

2. Lack satisfying solution

- online RL should be more efficient and effective than offline setting
- need for residual policies, reducing to latent space seem unsatisfying compared to directly performing RL on VLA weights

Next time

Final lecture: Summary, open problems, how to do RL research

Course reminders

- Poster session next Wednesday
- Final report due following Monday

^ no extensions or late days